High Performance Machine Learning Lab 3

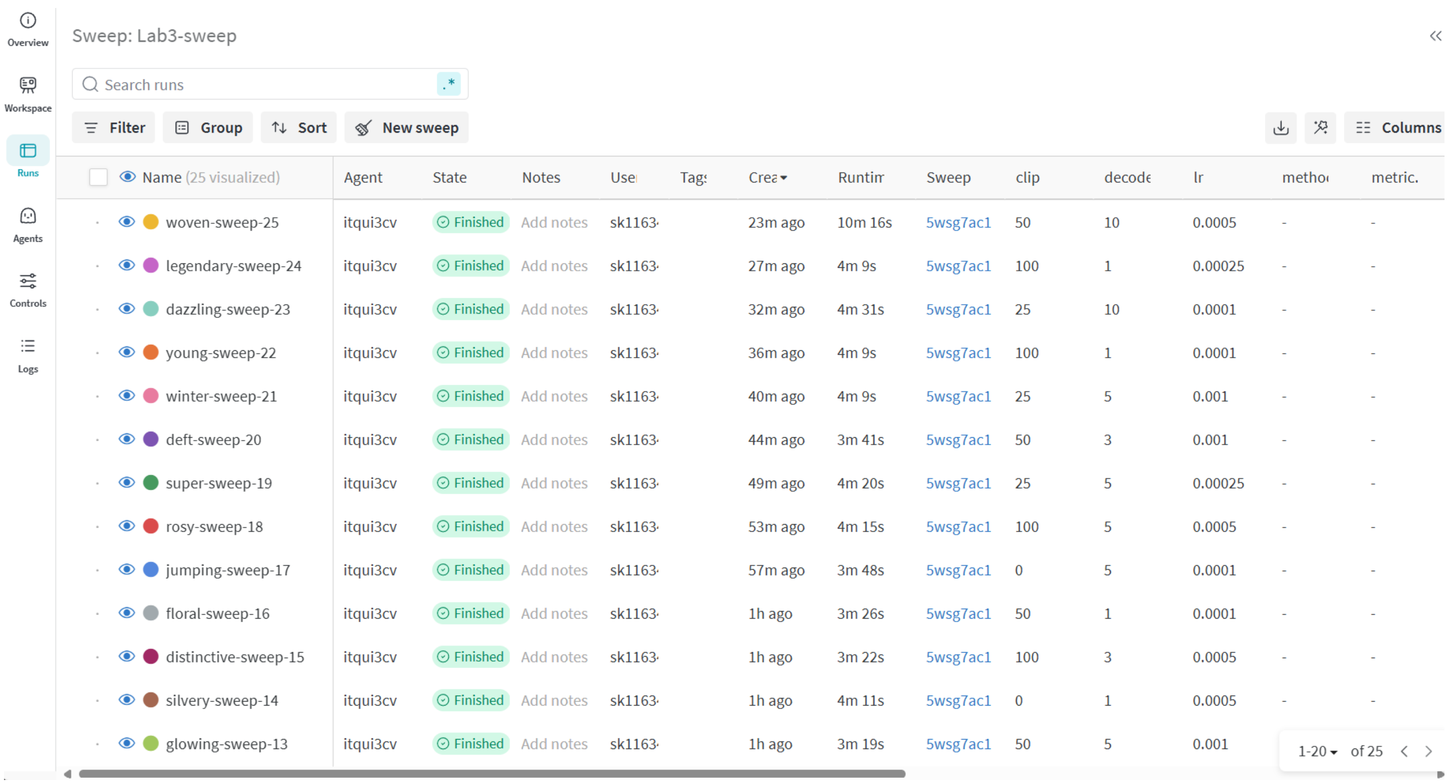
Name: Sarang P. Kadakia
NYU ID: sk11634

Problem 1: Chatbot Seq-2-Seq Model

1. **Wandb Project:** https://wandb.ai/sk11634-new-york-university/HPML_HW3_Sarang?nw=nwusersk11634

2. **Hyperparameter Sweeps:**

1.1. Strategy: Random Search
Total Runs: 25

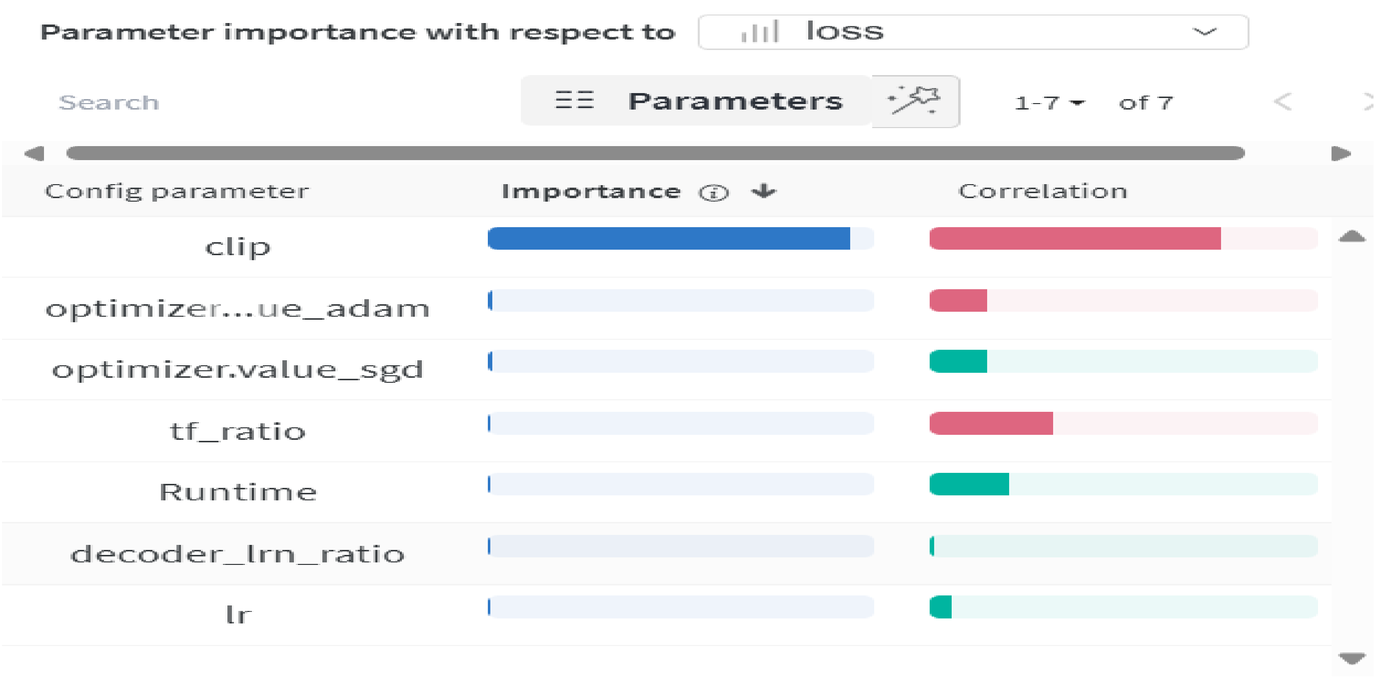


Name (25 visualized)	Agent	State	Notes	Use	Tag:	Crea	Runtir	Sweep	clip	decode	lr	metho	metric.
woven-sweep-25	itqui3cv	Finished	Add notes	sk1163		23m ago	10m 16s	5wsg7ac1	50	10	0.0005	-	-
legendary-sweep-24	itqui3cv	Finished	Add notes	sk1163		27m ago	4m 9s	5wsg7ac1	100	1	0.00025	-	-
dazzling-sweep-23	itqui3cv	Finished	Add notes	sk1163		32m ago	4m 31s	5wsg7ac1	25	10	0.0001	-	-
young-sweep-22	itqui3cv	Finished	Add notes	sk1163		36m ago	4m 9s	5wsg7ac1	100	1	0.0001	-	-
winter-sweep-21	itqui3cv	Finished	Add notes	sk1163		40m ago	4m 9s	5wsg7ac1	25	5	0.001	-	-
deft-sweep-20	itqui3cv	Finished	Add notes	sk1163		44m ago	3m 41s	5wsg7ac1	50	3	0.001	-	-
super-sweep-19	itqui3cv	Finished	Add notes	sk1163		49m ago	4m 20s	5wsg7ac1	25	5	0.00025	-	-
rosy-sweep-18	itqui3cv	Finished	Add notes	sk1163		53m ago	4m 15s	5wsg7ac1	100	5	0.0005	-	-
jumping-sweep-17	itqui3cv	Finished	Add notes	sk1163		57m ago	3m 48s	5wsg7ac1	0	5	0.0001	-	-
floral-sweep-16	itqui3cv	Finished	Add notes	sk1163		1h ago	3m 26s	5wsg7ac1	50	1	0.0001	-	-
distinctive-sweep-15	itqui3cv	Finished	Add notes	sk1163		1h ago	3m 22s	5wsg7ac1	100	3	0.0005	-	-
silvery-sweep-14	itqui3cv	Finished	Add notes	sk1163		1h ago	4m 11s	5wsg7ac1	0	1	0.0005	-	-
glowing-sweep-13	itqui3cv	Finished	Add notes	sk1163		1h ago	3m 19s	5wsg7ac1	50	5	0.001		

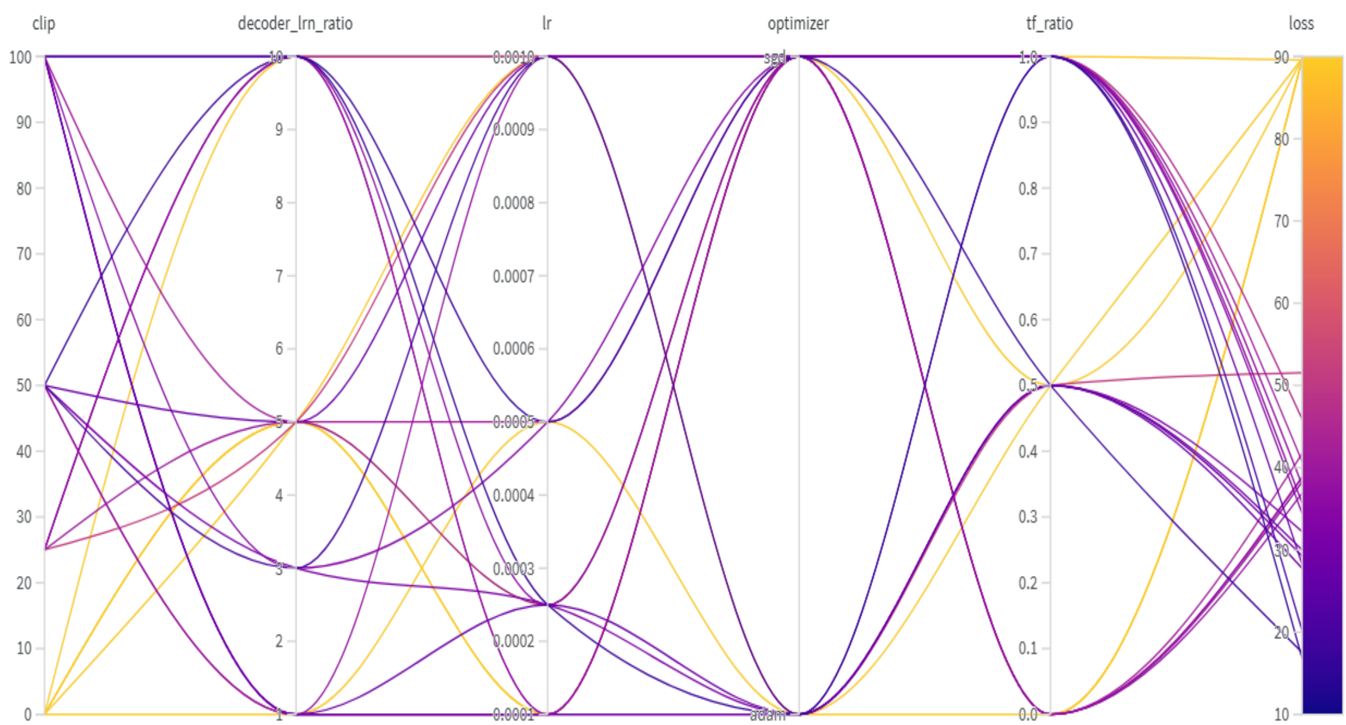
From above results, I observed that run with id sk11634 has least loss of *2.3088*. Hyperparameters value of above model:

1. Optimizer: adam
2. Learning Rate: 0.00025
3. Clip: 100.0
4. Teacher Forcing Ratio: 1
5. Decoder Learning Ratio: 10

3. Feature Importance:



From the panel above, we can see that the clip parameter has a significant impact on the loss. A higher clip value results in lower loss, as indicated by its negative correlation. Additionally, both ADAM and SGD optimizers show a strong correlation with loss, but ADAM has a positive correlation, making it the preferable choice. The importance metric for clip is considerably higher than other hyperparameters, suggesting that if the clip value is too low, the loss will remain high regardless of other parameter values, as evident from the diagram.



4. Variation of loss with iterations

There were 4000 iterations for each run!

5. PyTorch Profiler Tracing

I saved the 'Trace file' as *trace.json*

6. Measurement of time and memory Consumption of model's operators

This is my profiler.txt file:

Name	Self CPU	CPU total	CPU time avg	CUDA total	CUDA time avg	CPU Mem	Self CPU Mem	CUDA Mem	Self CUDA Mem
aten::empty	523.000us	523.000us	6.226us	0.000us	0.000us	24 b	24 b	335.39 Mb	335.39 Mb
aten::embedding	4.636ms	6.319ms	574.455us	56.000us	5.091us	0 b	0 b	24.00 Kb	0 b
aten::reshape	27.000us	41.000us	3.727us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::view	31.000us	31.000us	1.292us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::index_select	1.028ms	1.631ms	148.273us	56.000us	5.091us	0 b	0 b	24.00 Kb	0 b
aten::resize_	99.000us	99.000us	9.000us	0.000us	0.000us	0 b	0 b	24.00 Kb	24.00 Kb
cudaLaunchKernel	33.995ms	33.995ms	157.384us	209.000us	0.968us	0 b	0 b	0 b	0 b
ous namespace)::indexSelectS...	0.000us	0.000us	0.000us	56.000us	5.091us	0 b	0 b	0 b	0 b
aten::to	3.000us	3.000us	3.000us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::_pack_padded_sequence	51.000us	964.000us	964.000us	3.000us	3.000us	16 b	0 b	4.00 Kb	0 b
aten::slice	201.000us	210.000us	16.154us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::as_strided	69.000us	69.000us	0.476us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::cat	1.278ms	1.735ms	55.968us	133.000us	4.290us	0 b	0 b	54.00 Kb	54.00 Kb
aten::narrow	7.000us	16.000us	16.000us	0.000us	0.000us	0 b	0 b	0 b	0 b
cudaMemcpyAsync	49.000us	49.000us	24.500us	0.000us	0.000us	0 b	0 b	0 b	0 b
Memcpy DtoD (Device -> Device)	0.000us	0.000us	0.000us	6.000us	3.000us	0 b	0 b	0 b	0 b
aten::select	101.000us	113.000us	5.136us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::item	8.000us	10.000us	5.000us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::_local_scalar_dense	4.000us	4.000us	2.000us	0.000us	0.000us	0 b	0 b	0 b	0 b
aten::zeros	27.000us	1.613ms	537.667us	2.000us	0.667us	0 b	0 b	8.00 Kb	0 b

---

Problem 2: TorchScript Seq-2-Seq Model

Q1. Explain the differences between tracing and scripting and how they are used in TorchScript?

Solution:

Tracing: In tracing, the function takes both the model and sample input data to record computations and construct a graph-based function. However, it only captures the computations performed for the given input, making it incapable of handling data-dependent control flow.

Scripting: In scripting, only the model is required, without the need for sample data. This approach converts the model code into TorchScript, a subset of Python that includes all control flows.

For models without control flow dependencies, torch.jit.trace() can be used without modifying the model, as it directly converts it into TorchScript. However, for scripting, the model code may need adjustments to conform to TorchScript syntax before using torch.jit.script(). When using tracing, it is necessary to set the model's device and dropout layers to test mode before tracing, as the traced model does not inherently handle these operations. In contrast, scripting allows setting the device and dropout layers to test mode just before inference, similar to how it's done in eager mode.

Q2. Explain the changes needed in the chatbot model to allow for scripting.

Solution:

In the given model, there are three sub-modules: Encoder, Decoder, and *GreedySearchDecoder*. The third sub-module, *GreedySearchDecoder*, requires scripting due to its input-based control flow.

Changes required in *GreedySearchDecoder*:

1. **Passing decoder_n_layers as a Constructor Argument:** Earlier, it was using fetching this value from decoder but since we are using traced version of decoder we will not be able to access that value anymore and hence we need to pass this to its constructor for it to use.

```
#Modified GreedySearchDecoder for scripting the module

class GreedySearchDecoderScript(torch.jit.ScriptModule):
    def __init__(self, encoder, decoder, decoder_n_layers):
```

2. **Explicit Type Annotations for Forward Method Arguments:** By default, TorchScript assume all parameters of function as tensor. Hence, in case we need to pass any argument with different type like int in our case, we need to specify the type in python function.

3. **Adding More Attributes to Handle Global Variables:** Earlier, we were accessing various variable available in global scope of our python environment, but TorchScript version do not have access to those variables, and we need to save value of those variables from global scope to attributes of the model class. *(_SOS_token, _device, _decoder_n_layers)*

```
self._device = device
self._SOS_token = SOS_token
self._decoder_n_layers = decoder_n_layers
```

Q3: Comparing Latency (displayed as latency table):

	Latency on CPU (ms)	Latency on GPU (ms)
Pytorch	305.217959	11.282072
Torchscript	21.746694	14.643587
SpeedUp	14.035143	0.770445