

Homework-01: Introduction to AI for Systems

CSC 591: Foundations of Generative AI for Systems Spring 2025

Kahaan Patel - kpatel48

February 28th, 2025

Section 1 - Conceptual Questions

Question I

The authors mention two approaches to improve cache performance (hit rate). Which approach does the paper focus on?

Answer:

Cache performance is enhanced either by prefetching data into the cache prior to demands being made, or through selecting the optimal cache line to evict on a miss. This paper focuses on the second aspect: modifying the cache replacement policy. The authors formulate the eviction decision as an imitation learning task, where they train a model to do best as if it was supplied with Belady's oracle, which makes the perfect eviction decision when future accesses are known. This method seeks to increase the cache hit rate without requiring any sort of data prediction or prefetching.

Question II

What are the previous state-of-the-art approaches mentioned in the paper?

Answer:

The paper mentions Hawkeye and Glider as the previous best approaches to cache replacement. Both approaches exploit Belady's optimal policy by initially partitioning cache lines into two groups: cache-friendly and cache-averse. Then, they heuristically select a cache line to evict. While these strategies use oracle resources from Belady's algorithm, they are bounded by their reliance on simple forms of classification and crude heuristics instead of attempting to learn a comprehensive eviction policy as this paper suggests.

Question III

Explain the cache hierarchy used for experimentation by the authors. This means you have to give the size of L1/L2/L3 caches including the number of sets and ways.

Answer:

The studies are performed on a hierarchy of three caches. The L1 cache has a capacity of 32 KB and is 4-way set associative, and for a cache line of 64 bytes it has 128 sets. The L2 cache has 256 KB of capacity and is 8-way set associative, hence it has 512 sets. Finally, the L3 cache, which has 2 MB of cache is 16-way set associative and has 2048 sets. This distribution of size, a number of sets and their associativity presents good testbed for the effectiveness of multiple cache replacement algorithms.

Question IV

What is the normalized cache hit rate? Why have the authors used it as a metric?

Answer:

The normalized cache hit rate is defined as

$$\frac{r - r_{\text{LRU}}}{r_{\text{Belady}} - r_{\text{LRU}}},$$

where r is the hit rate achieved by a given policy, r_{LRU} is the hit rate for the Least Recently Used policy, and r_{Belady} is the hit rate of the optimal Belady’s policy. This metric measures the relative improvement of a policy over LRU compared to the optimal performance. It is used because it provides a standardized way to compare different policies by showing how much of the performance gap between a commonly used baseline LRU and the theoretical optimum is closed by the learned policy.

Question V

Interpret Figure 2 from the paper. Explain what you understand by looking at the figure and the caption.

Answer:

Figure 2 plots the normalized cache hit rate of Belady’s policy as a function of the number of future accesses it is allowed to observe. The graph shows that as the future window increases, the hit rate approaches the optimal value, demonstrating that access to more future information significantly improves eviction decisions. However, it also reveals that a substantial amount of future lookahead on the order of thousands of accesses is needed to approximate the full performance of Belady’s policy. This observation underscores the challenge of replicating Belady’s optimal behavior when only historical data is available, thereby motivating the need for a learned approximation.

Question VI

The following table shows the number of misses encountered while evaluating/testing the Parrot model in the second column. The last column estimates the total number of cache misses, i.e., the speculated number of cache misses for the complete trace and all cache sets. What formula might be used to calculate the values in the last column? Explain your answer.

Answer:

Cache misses can be estimated by determining the number of accesses that do not result in a hit. Since the cache hit rate represents the fraction of accesses that successfully retrieve data from the cache, the number of hits can be found by multiplying the total accesses by the hit rate. Subtracting this value from the total accesses gives the number of misses. The formula is:

$$\text{CacheMisses} = \text{TotalAccesses} - (\text{CacheHitRate} \times \text{TotalAccesses}).$$

For example, if a system processes 2000 memory accesses and achieves a 90% hit rate, the number of misses is

$$2000 - (0.90 \times 2000) = 200.$$

This calculation helps assess how often data is not found in the cache, highlighting the effectiveness of a replacement policy.

Section 2 - Belady and LRU Policies

Task 1 - LRU

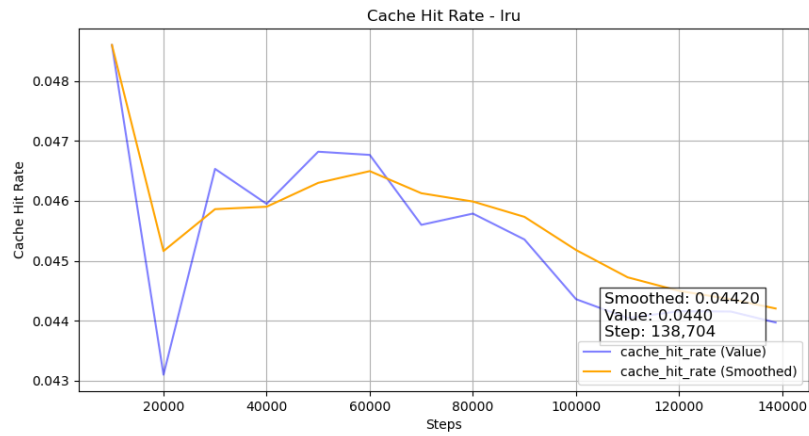


Figure 1: Cache Hit Rate for LRU Policy

Task 2 - Belady

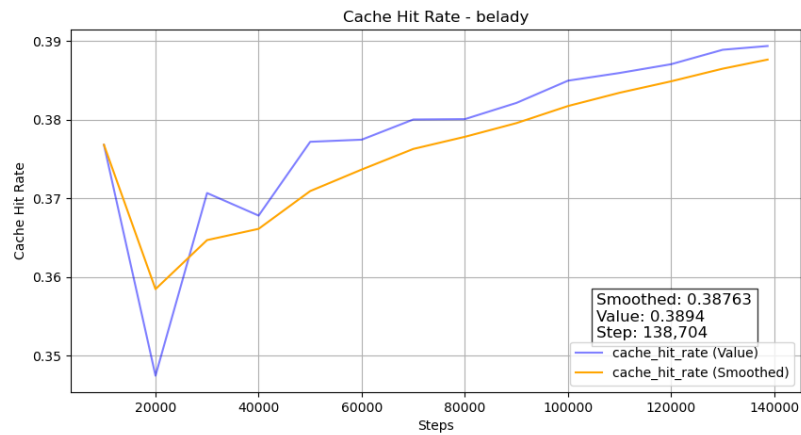


Figure 2: Cache Hit Rate for Belady's Optimal Policy

Section 3 - MLP

Task 1 - Layer Width

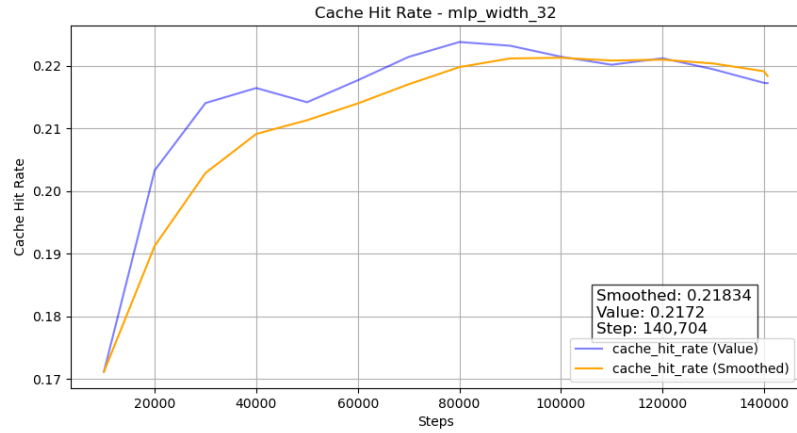


Figure 3: Effect of Layer Width 32 on Cache Hit Rate

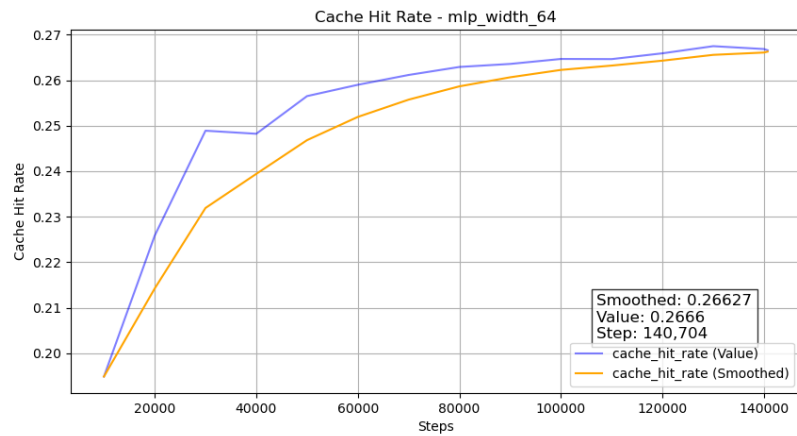


Figure 4: Effect of Layer Width 64 on Cache Hit Rate



Figure 5: Effect of Layer Width 128 on Cache Hit Rate



Figure 6: Effect of Layer Width 256 on Cache Hit Rate

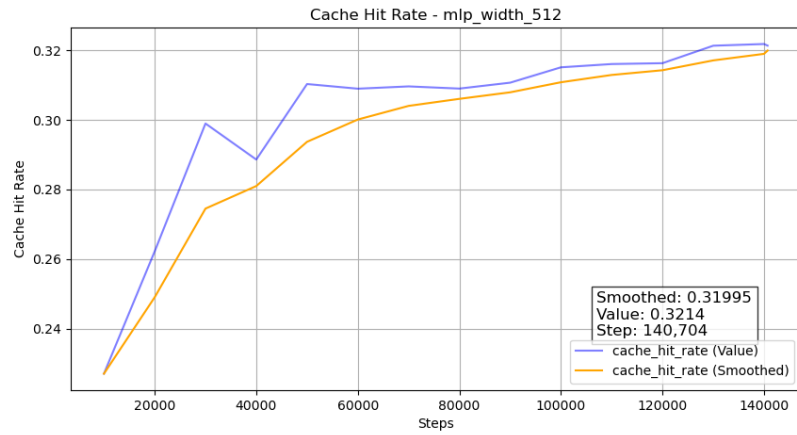


Figure 7: Effect of Layer Width 512 on Cache Hit Rate

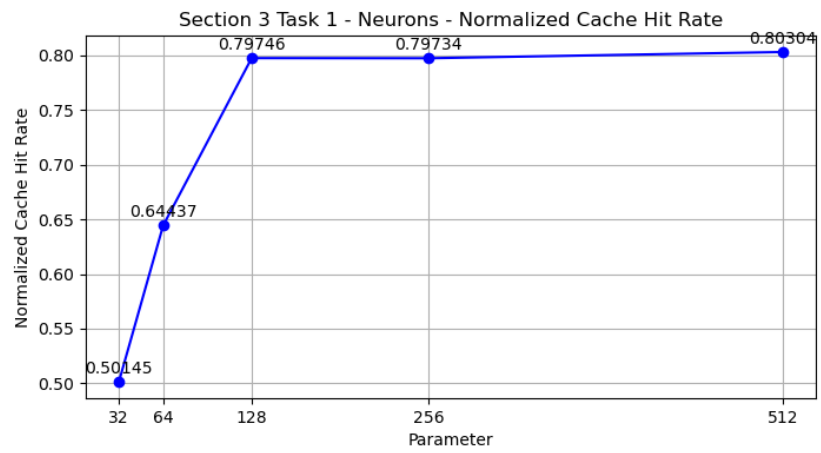


Figure 8: Effect of Number of Neurons on Normalized Cache Hit Rate

Task 2 - Number of Layers

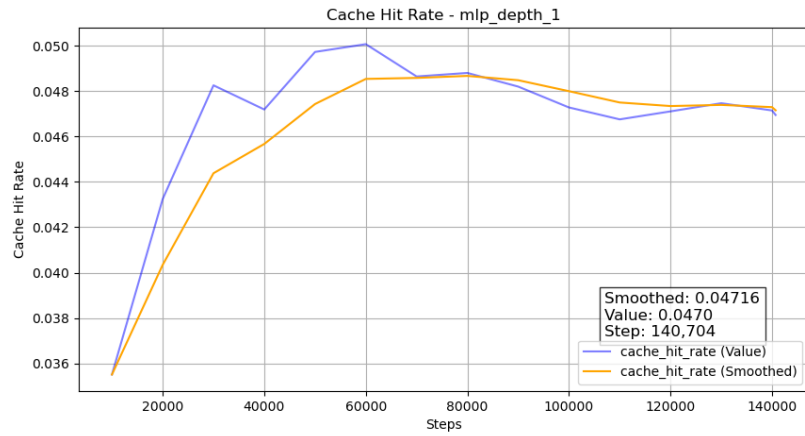


Figure 9: Effect of 1 Layer on Cache Hit Rate

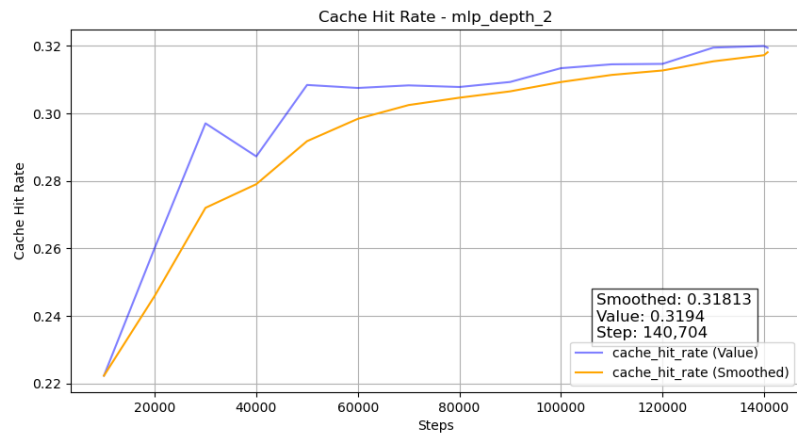


Figure 10: Effect of 2 Layers on Cache Hit Rate

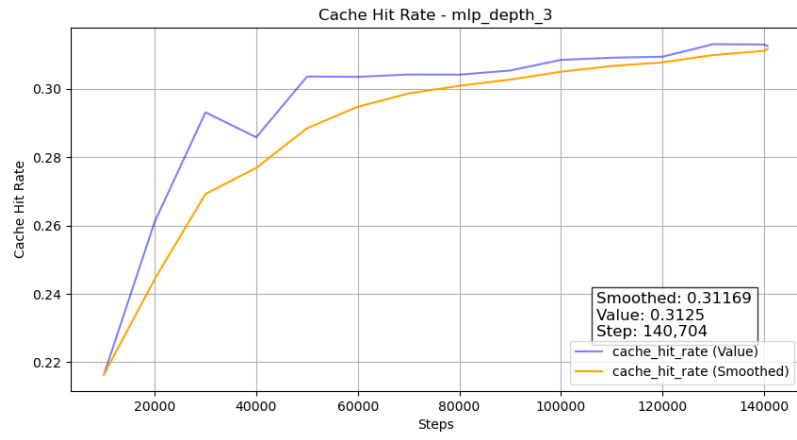


Figure 11: Effect of 3 Layers on Cache Hit Rate

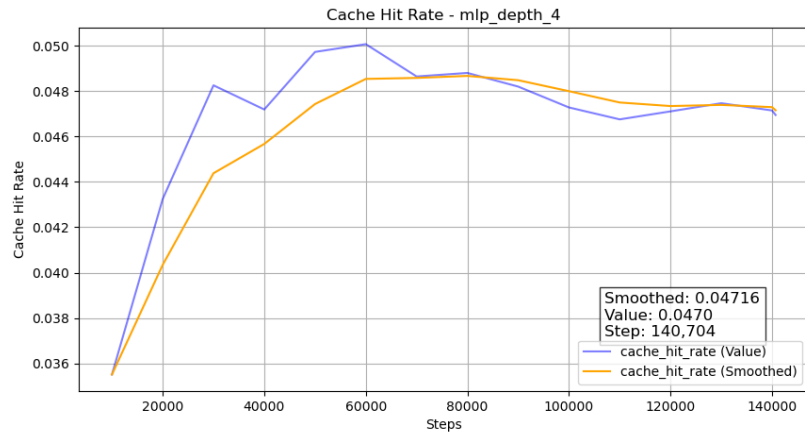


Figure 12: Effect of 4 Layers on Cache Hit Rate

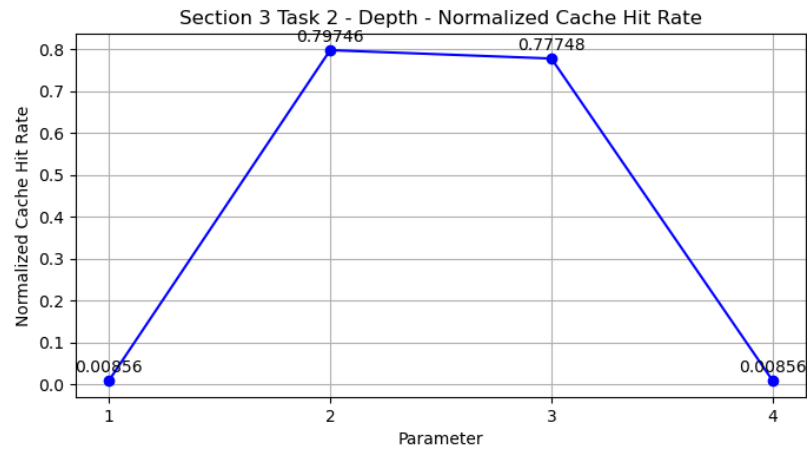


Figure 13: Effect of Depth on Normalized Cache Hit Rate

Task 3 - Activation Functions

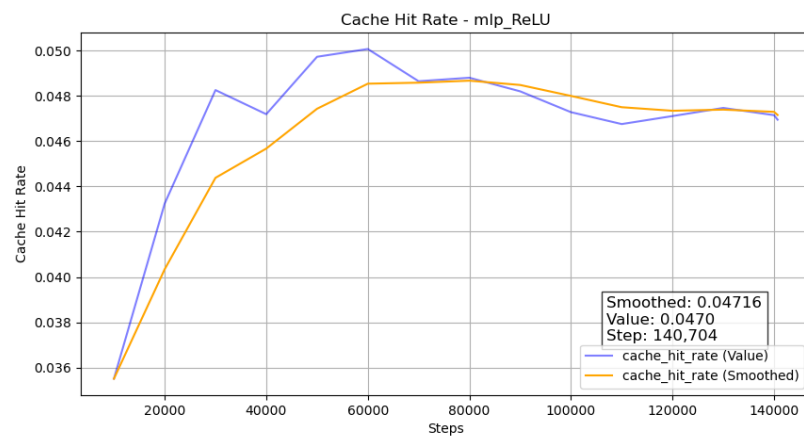


Figure 14: Effect of ReLU Activation on Cache Hit Rate

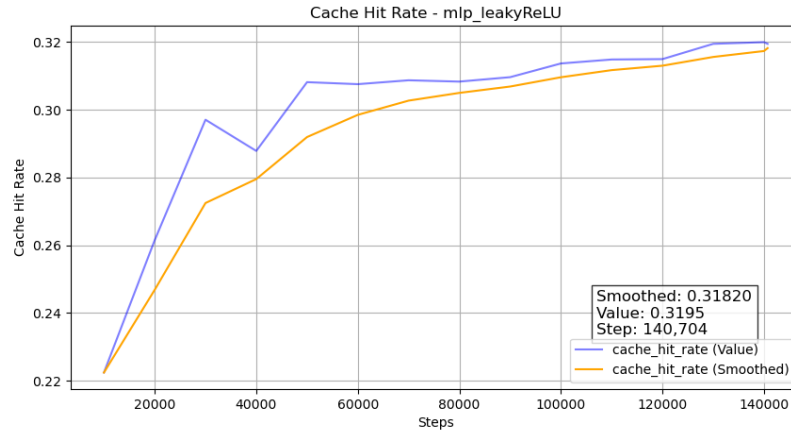


Figure 15: Effect of Leaky ReLU Activation on Cache Hit Rate

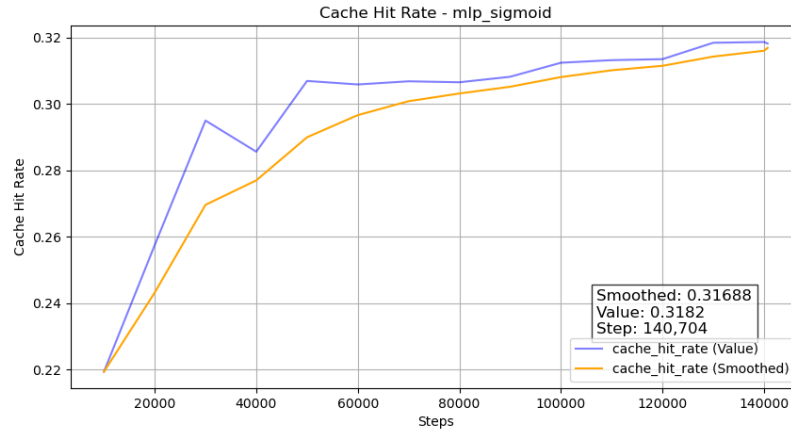


Figure 16: Effect of Sigmoid Activation on Cache Hit Rate

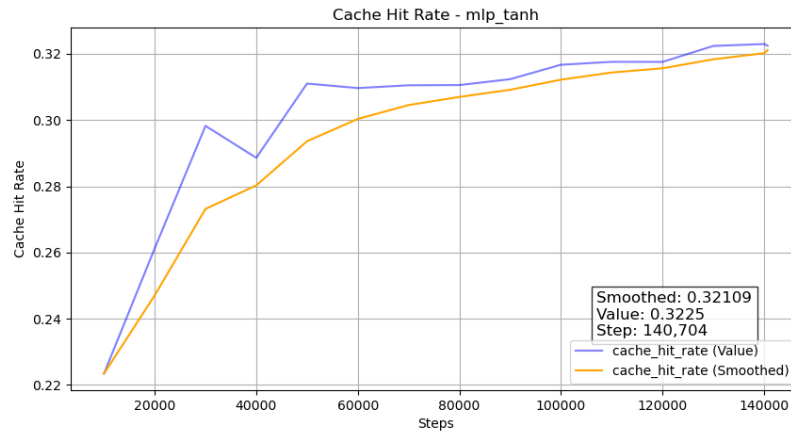


Figure 17: Effect of Tanh Activation on Cache Hit Rate

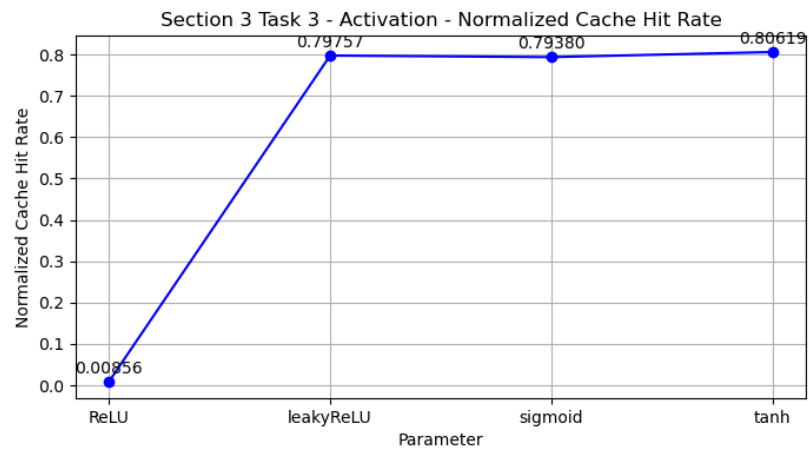


Figure 18: Effect of Activation Functions on Normalized Cache Hit Rate

Task 4 - Batch Size

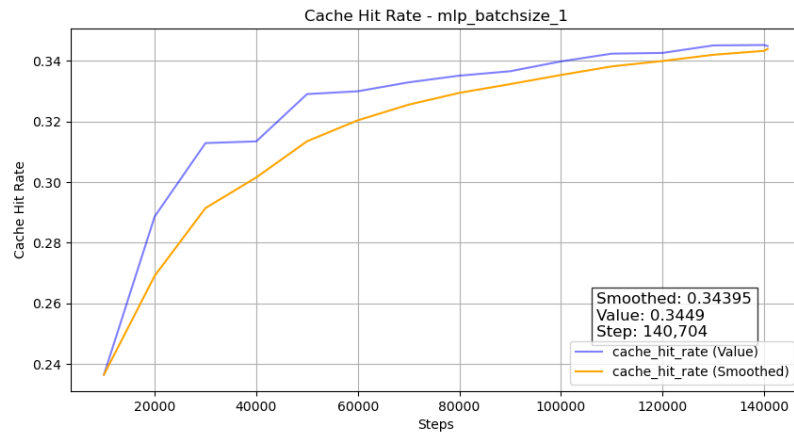


Figure 19: Effect of Batch Size 1 on Cache Hit Rate

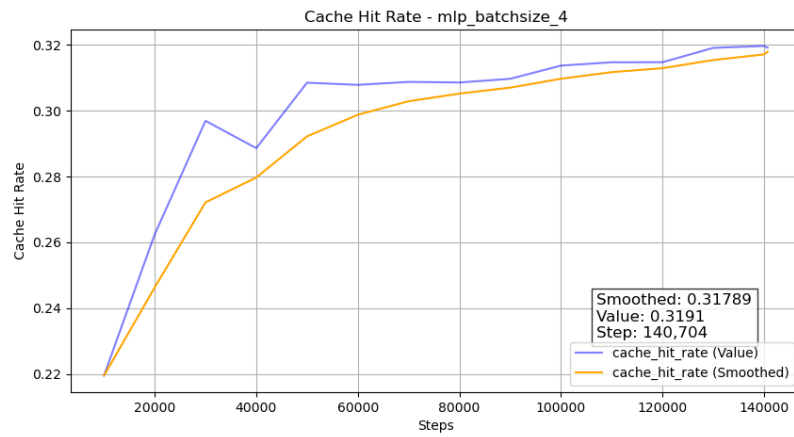


Figure 20: Effect of Batch Size 4 on Cache Hit Rate

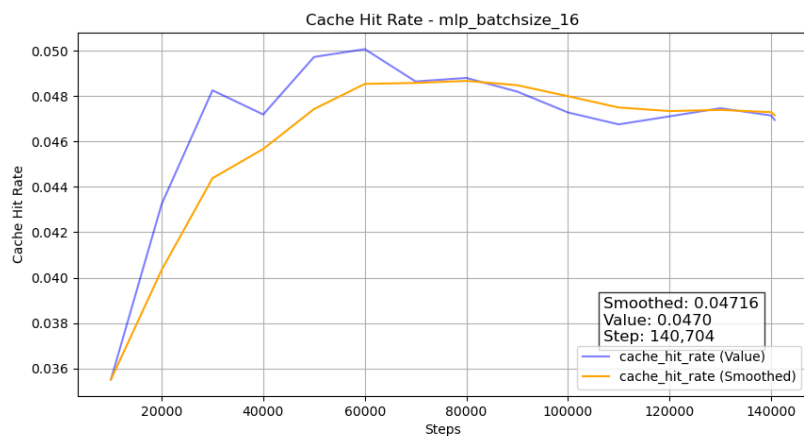


Figure 21: Effect of Batch Size 16 on Cache Hit Rate

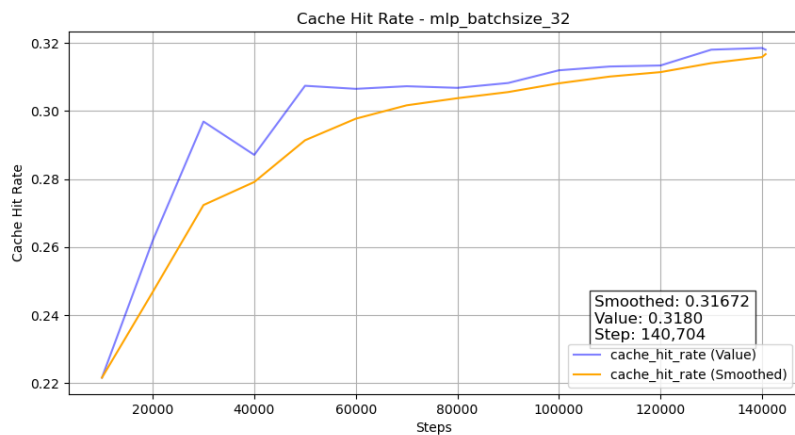


Figure 22: Effect of Batch Size 32 on Cache Hit Rate

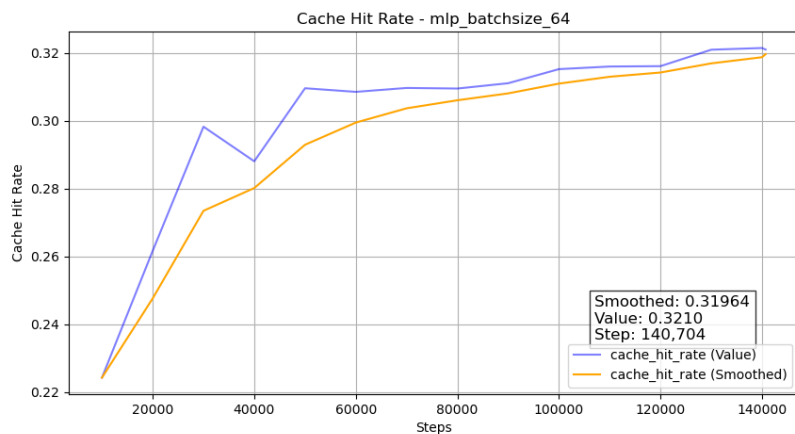


Figure 23: Effect of Batch Size 64 on Cache Hit Rate

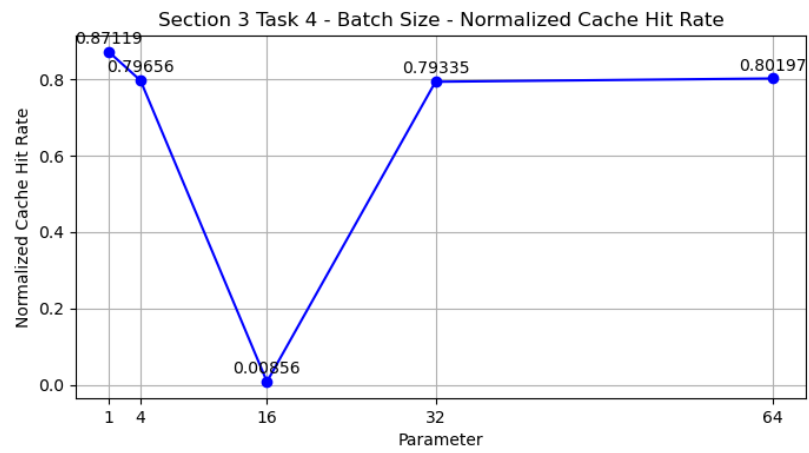


Figure 24: Effect of Batch Size on Normalized Cache Hit Rate

Task 5 - Learning Rates

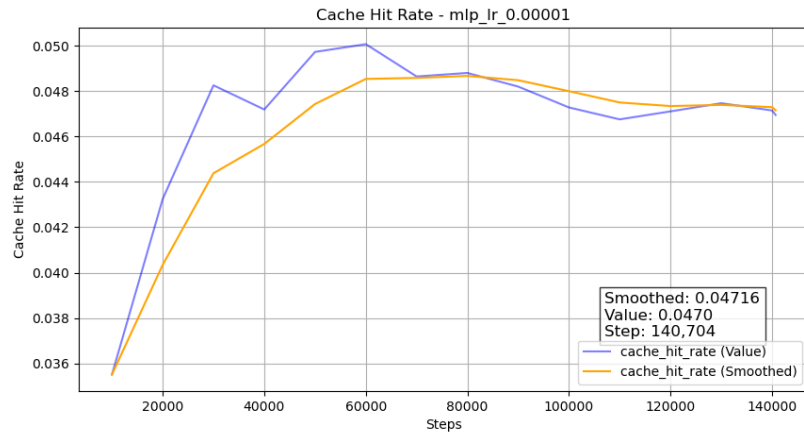


Figure 25: Effect of Learning Rate 0.00001 on Cache Hit Rate

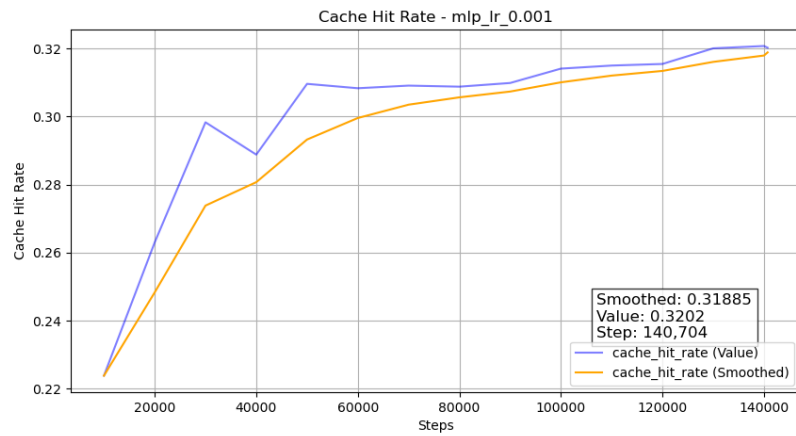


Figure 26: Effect of Learning Rate 0.001 on Cache Hit Rate

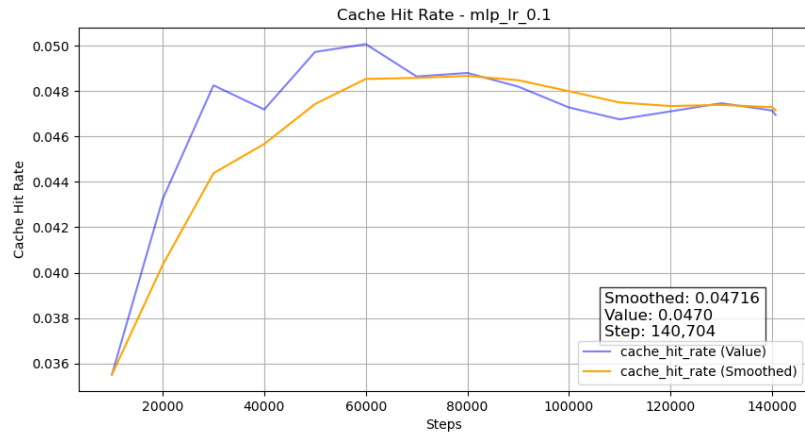


Figure 27: Effect of Learning Rate 0.1 on Cache Hit Rate

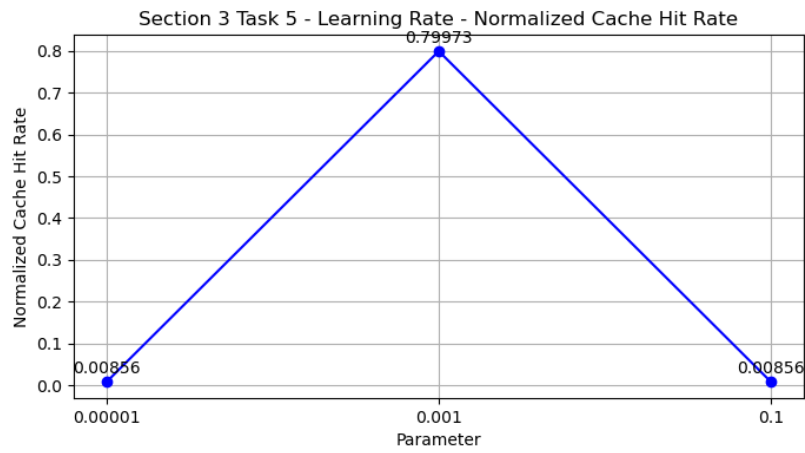


Figure 28: Effect of Learning Rate on Normalized Cache Hit Rate

Section 4 - RNN

Task 1 - Varying History Length With Attention

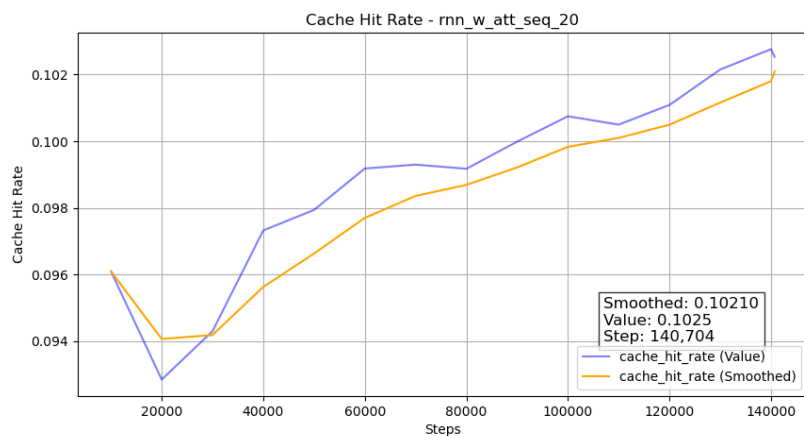


Figure 29: Effect of History Length 20 on Cache Hit Rate

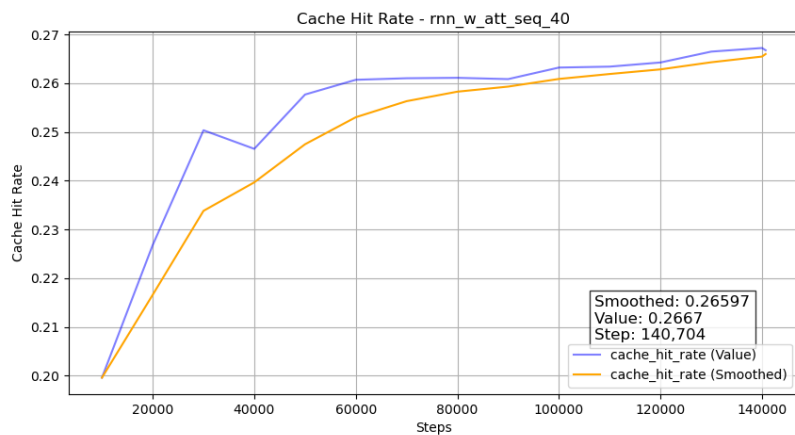


Figure 30: Effect of History Length 40 on Cache Hit Rate

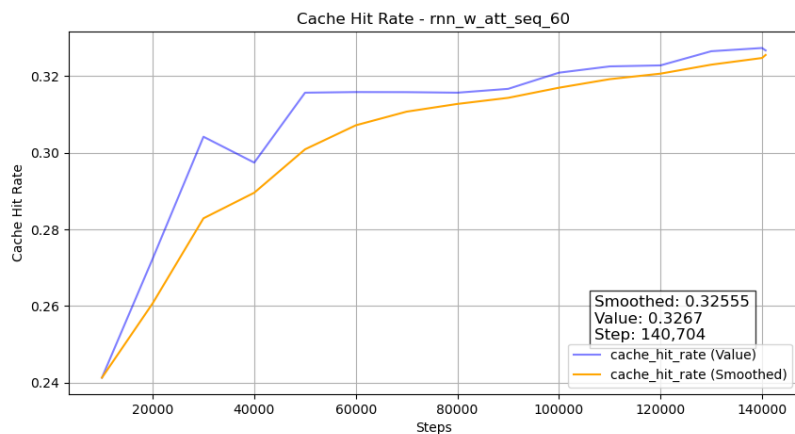


Figure 31: Effect of History Length 60 on Cache Hit Rate

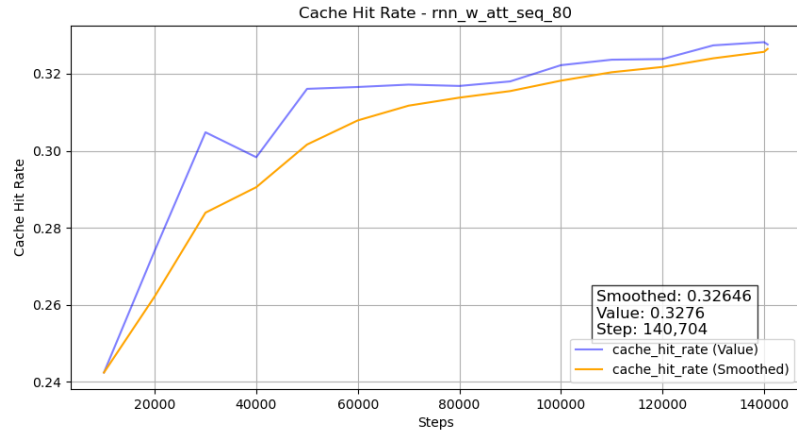


Figure 32: Effect of History Length 80 on Cache Hit Rate

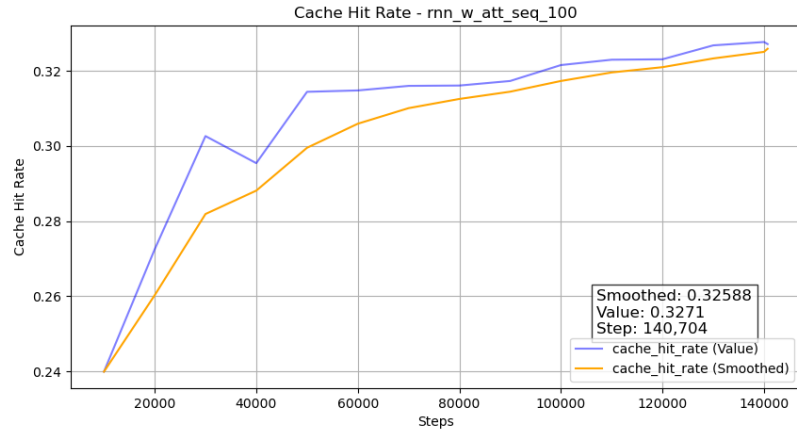


Figure 33: Effect of History Length 100 on Cache Hit Rate

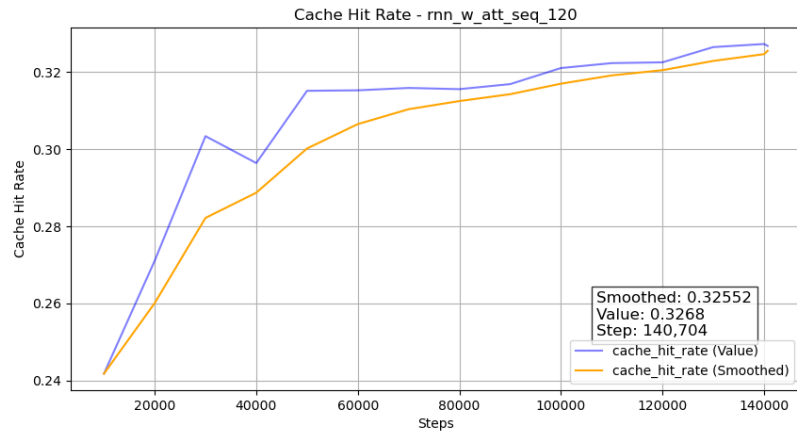


Figure 34: Effect of History Length 120 on Cache Hit Rate

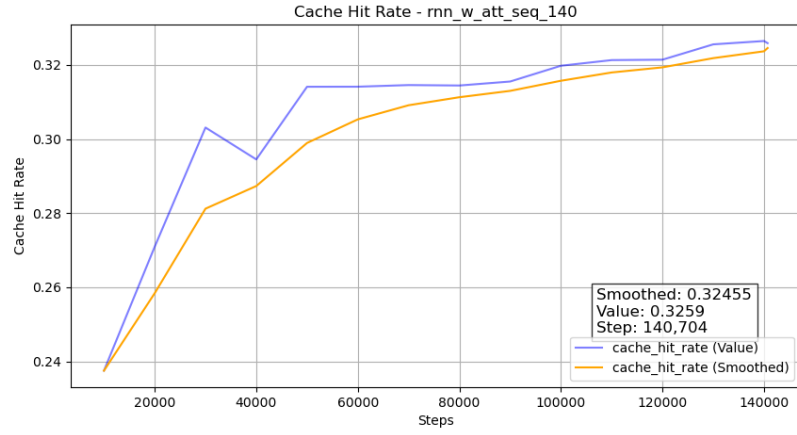


Figure 35: Effect of History Length 140 on Cache Hit Rate

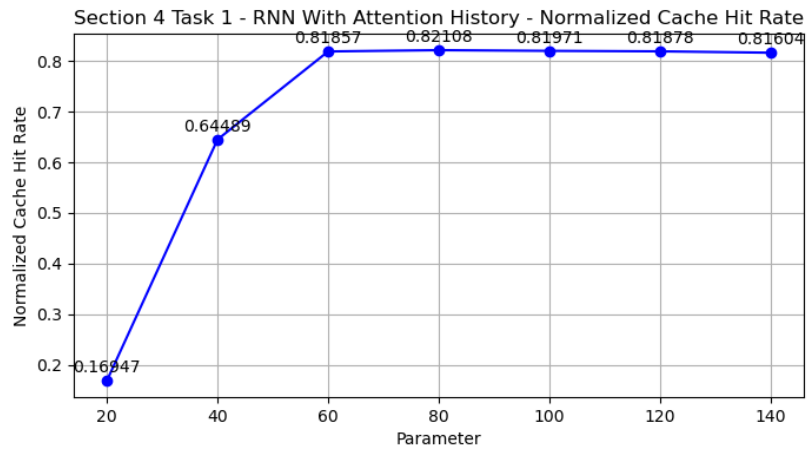


Figure 36: Effect of Attention History Length on Normalized Cache Hit Rate

Task 2 - Varying History Length Without Attention

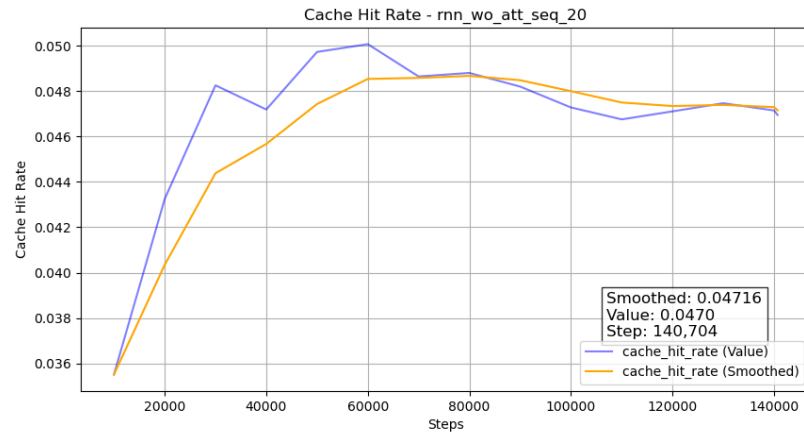


Figure 37: Effect of History Length 20 on Cache Hit Rate

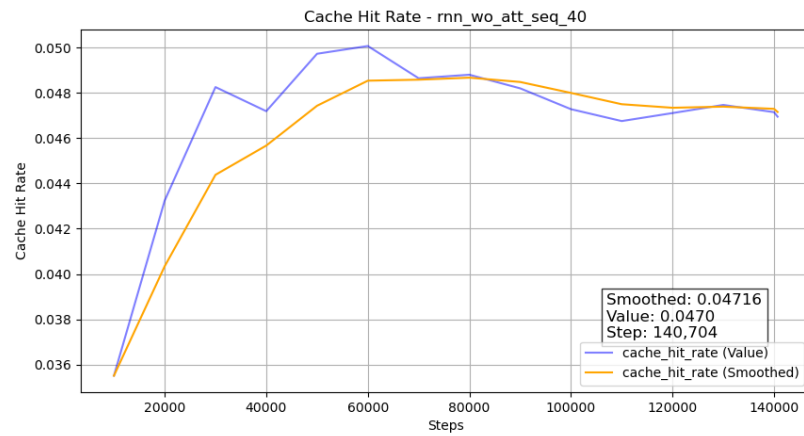


Figure 38: Effect of History Length 40 on Cache Hit Rate

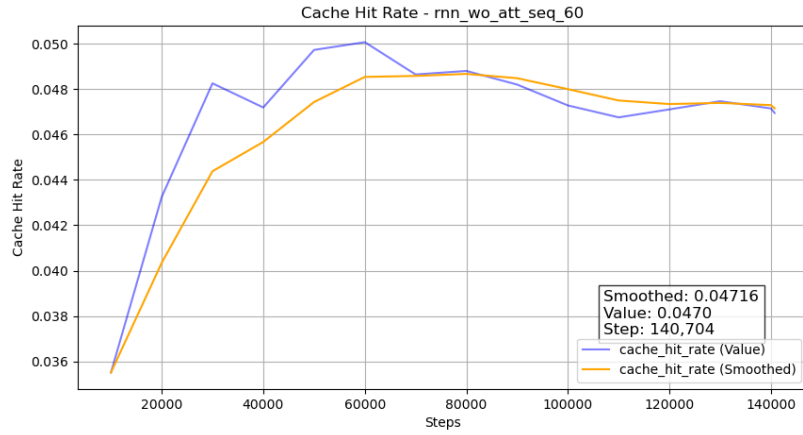


Figure 39: Effect of History Length 60 on Cache Hit Rate

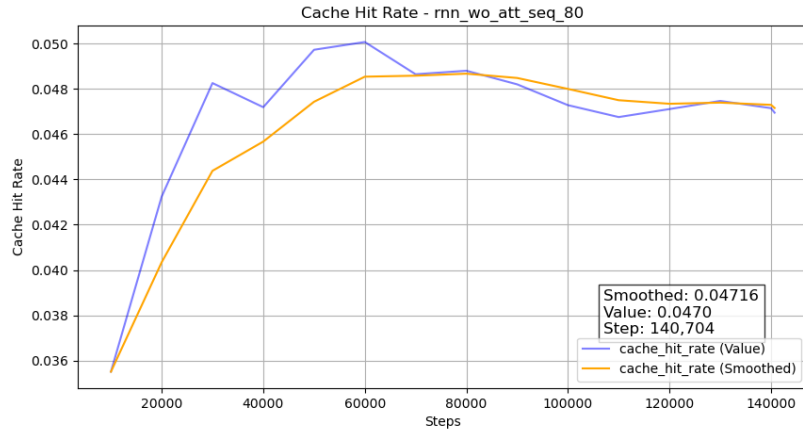


Figure 40: Effect of History Length 80 on Cache Hit Rate

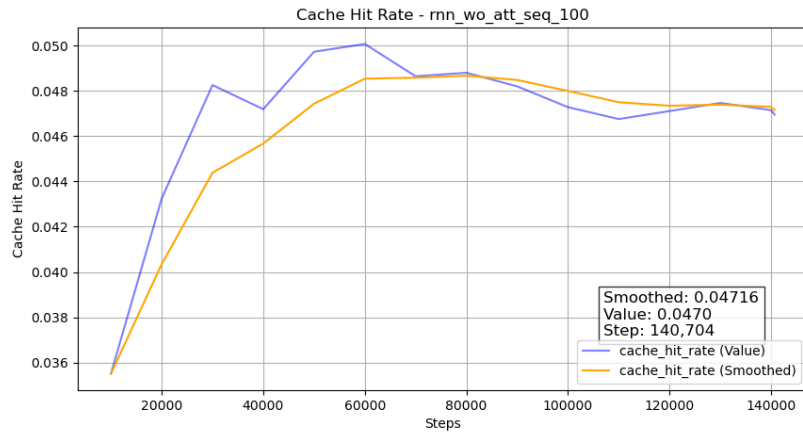


Figure 41: Effect of History Length 100 on Cache Hit Rate



Figure 42: Effect of History Length 120 on Cache Hit Rate

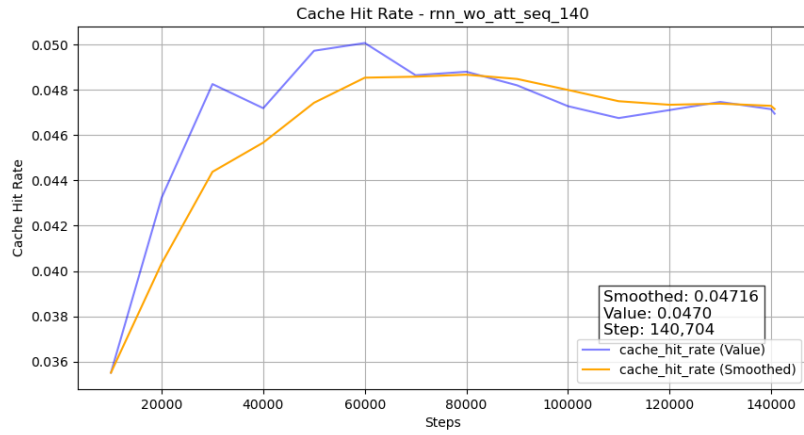


Figure 43: Effect of History Length 140 on Cache Hit Rate

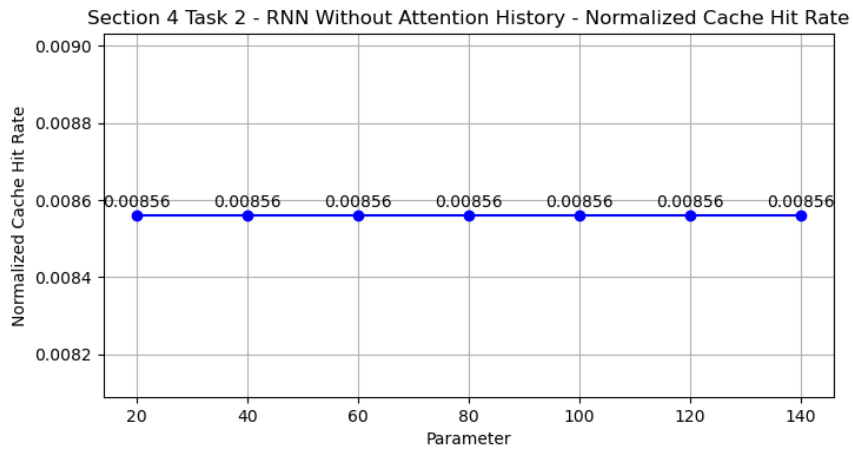


Figure 44: Effect of History Length on Normalized Cache Hit Rate (Without Attention)

Task 3 - Varying Max Attention History

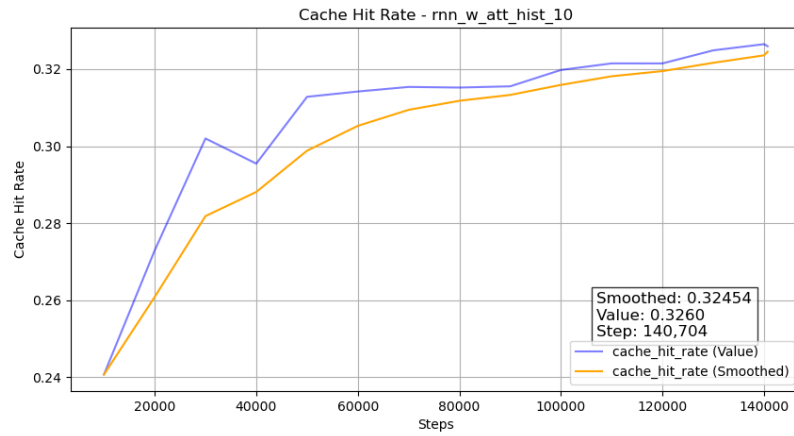


Figure 45: Effect of Max Attention History 10 on Cache Hit Rate

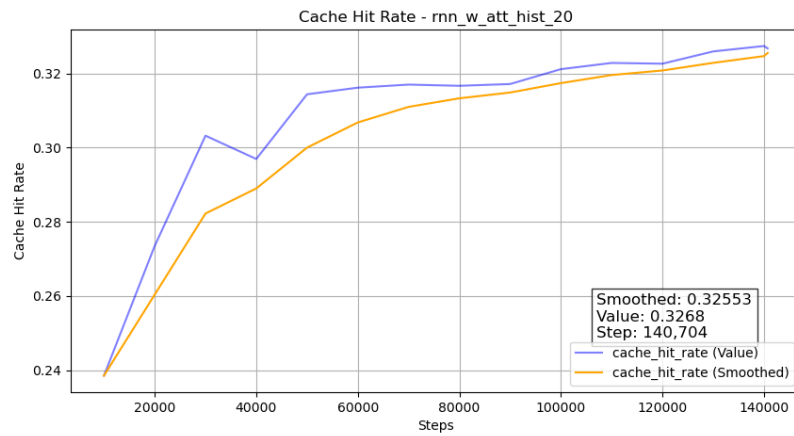


Figure 46: Effect of Max Attention History 20 on Cache Hit Rate

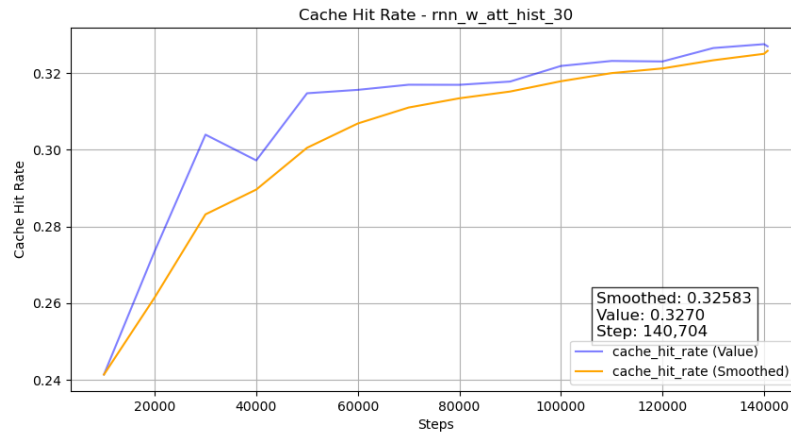


Figure 47: Effect of Max Attention History 30 on Cache Hit Rate

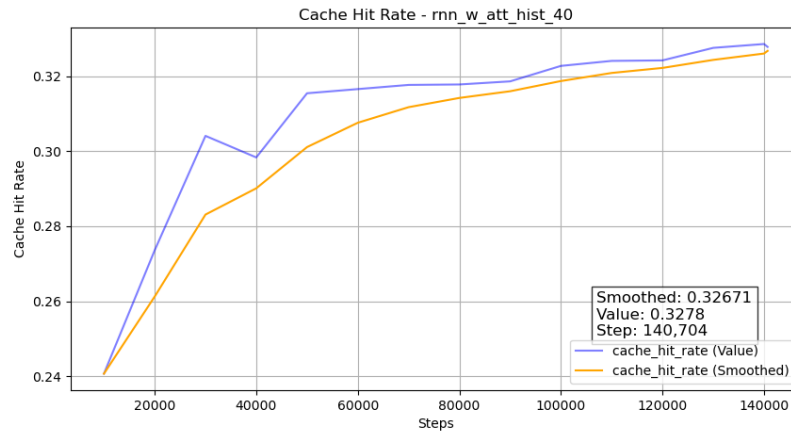


Figure 48: Effect of Max Attention History 40 on Cache Hit Rate

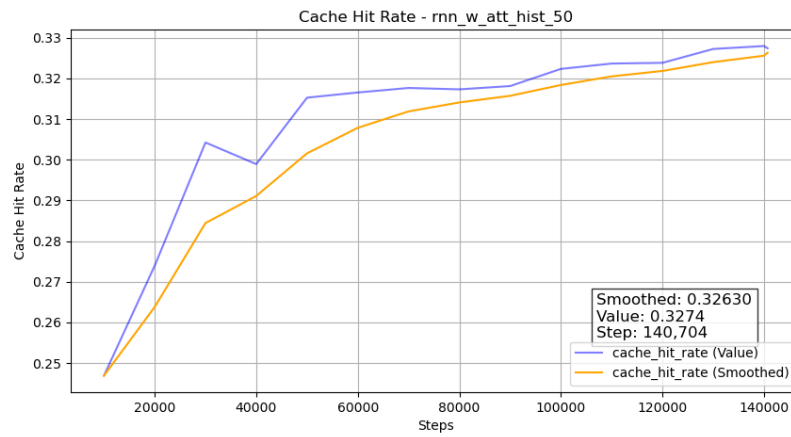


Figure 49: Effect of Max Attention History 50 on Cache Hit Rate

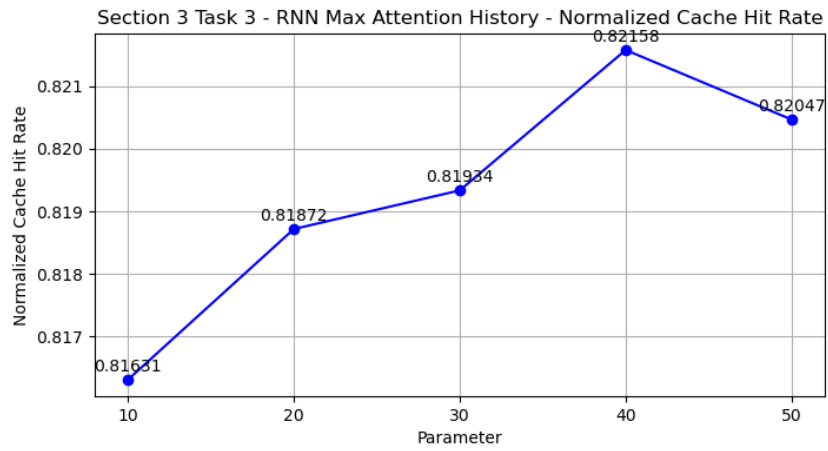


Figure 50: Effect of Max Attention History on Normalized Cache Hit Rate

Section 5 - LSTM

Task 1 - Baseline

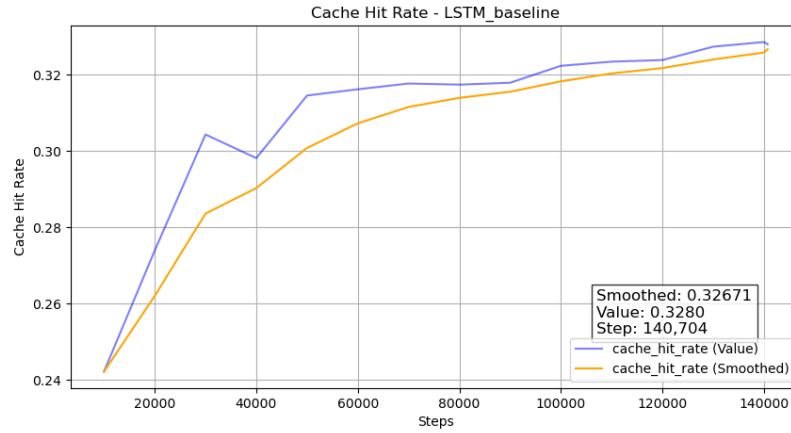


Figure 51: Cache Hit Rate for LSTM Baseline Model

Task 2 - Byte Embedder

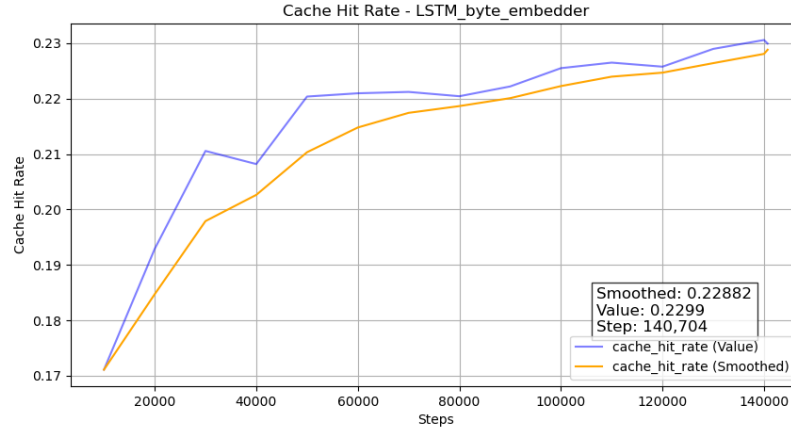


Figure 52: Cache Hit Rate for LSTM with Byte Embedder

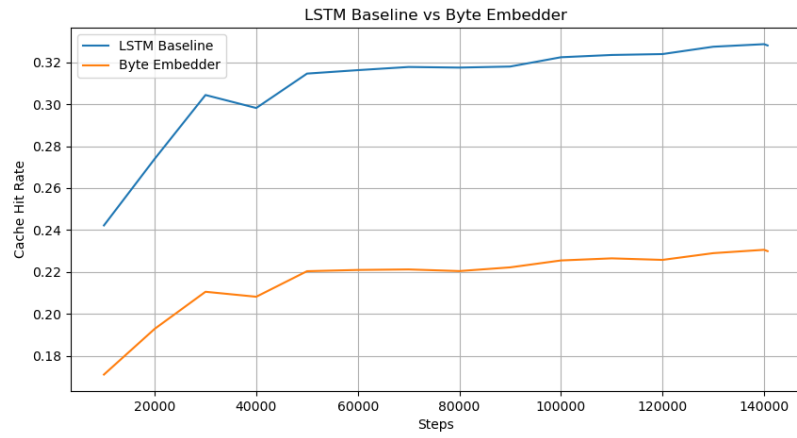


Figure 53: Cache Hit Rate for LSTM with Byte Embedder

Task 3 - Ablation - Reuse Distance Loss

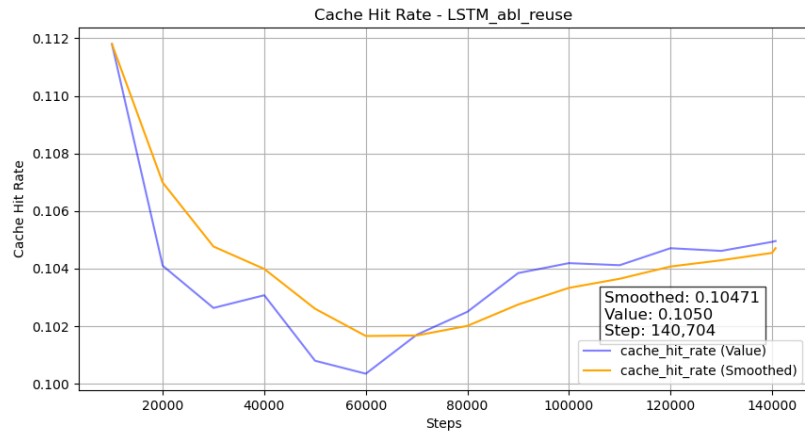


Figure 54: Effect of Removing Reuse Distance Loss on Cache Hit Rate

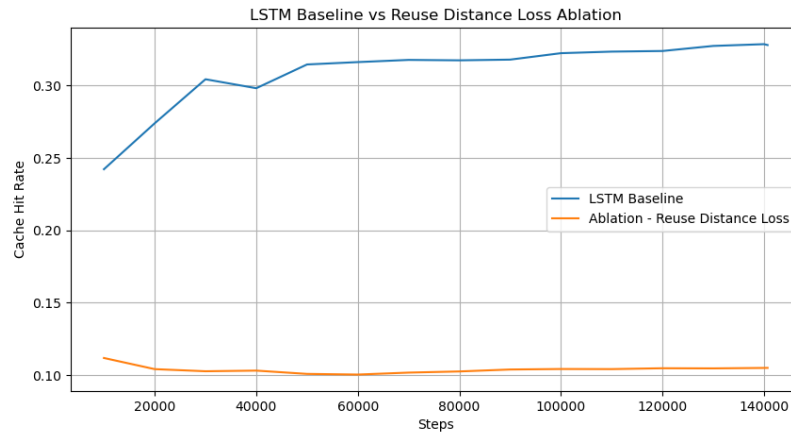


Figure 55: Effect of Removing Reuse Distance Loss on Cache Hit Rate

Task 4 - Ablation - Ranking Loss

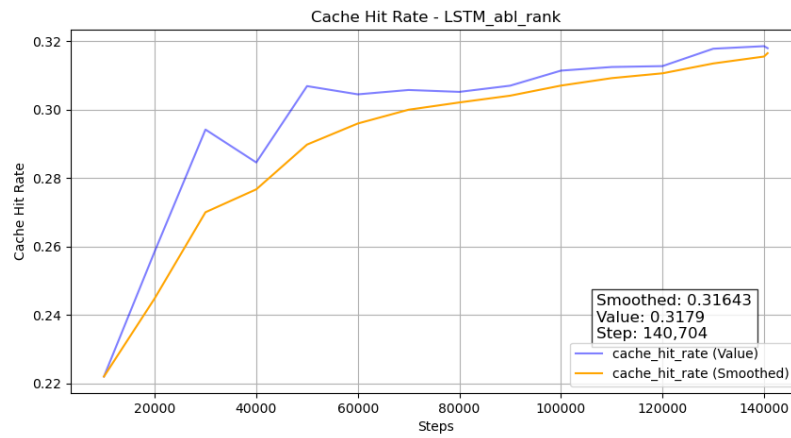


Figure 56: Effect of Removing Ranking Loss on Cache Hit Rate

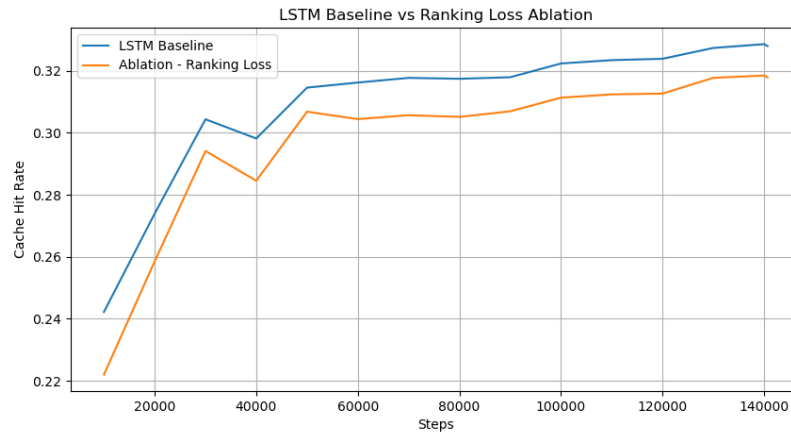


Figure 57: Effect of Removing Ranking Loss on Cache Hit Rate

GitHub Repository

The source code for this project, including implementations and scripts used for analysis, is available on GitHub. You can access the repository at: https://github.com/PatelKahaan/GenAI_HW01

Screenshots

The following section contains screenshots of the system tmp files, eval files and execution logs.

```

(base) [kpatel48@login02 tmp]$ cd ..
(base) [kpatel48@login02 kpatel48]$ ls
eval  GenAI-for-Systems-Gym  logs  tensorboard  tmp  tmp_eval  tmp_new  trf.sh
(base) [kpatel48@login02 kpatel48]$ cd logs
(base) [kpatel48@login02 logs]$ ls
LSTM_abl_rank.err.62088  mlp_depth_4.err.62062  mlp_width_256.err.61998  mlp_width_64.err.61999  rnn_w_att_seq_20.err.62064
LSTM_abl_rank.out.62088  mlp_depth_4.out.62062  mlp_width_256.err.62094  mlp_width_64.err.62045  rnn_w_att_seq_20.out.62064
LSTM_abl_reuse.err.62087  mlp_leakyReLU.err.62046  mlp_width_256.err.83063  mlp_width_64.err.83066  rnn_w_att_seq_40.err.62071
LSTM_abl_reuse.out.62087  mlp_leakyReLU.out.62046  mlp_width_256.eval.err.65952  mlp_width_64.eval.err.65953  rnn_w_att_seq_40.out.62071
LSTM_baseline.err.62083  mlp_lr_0.00001.err.62063  mlp_width_256.eval.out.65952  mlp_width_64.eval.out.65953  rnn_w_att_seq_60.err.62081
LSTM_baseline.out.62083  mlp_lr_0.00001.out.62063  mlp_width_256.out.61998  mlp_width_64.out.61999  rnn_w_att_seq_60.out.62081
LSTM_byte_embedder.err.62086  mlp_lr_0.001.err.62069  mlp_width_256.out.62054  mlp_width_64.out.62045  rnn_w_att_seq_80.err.62070
LSTM_byte_embedder.out.62086  mlp_lr_0.001.out.62069  mlp_width_256.out.83063  mlp_width_64.out.83066  rnn_w_att_seq_80.out.62070
mlp_batchsize_16.err.62060  mlp_lr_0.1.err.62065  mlp_width_32.err.61997  rnn_w_att_hist_10.err.62077  rnn_wo_att_seq_100.err.62076
mlp_batchsize_16.out.62060  mlp_lr_0.1.out.62065  mlp_width_32.err.62057  rnn_w_att_hist_10.out.62077  rnn_wo_att_seq_100.out.62076
mlp_batchsize_1.err.62056  mlp_ReLU.err.62052  mlp_width_32.err.83064  rnn_w_att_hist_20.err.62075  rnn_wo_att_seq_120.err.62074
mlp_batchsize_1.out.62056  mlp_ReLU.out.62052  mlp_width_32.eval.err.65954  rnn_w_att_hist_20.out.62075  rnn_wo_att_seq_120.out.62074
mlp_batchsize_32.err.62055  mlp_sigmoid.err.62061  mlp_width_32.eval.out.61997  rnn_w_att_hist_30.err.62085  rnn_wo_att_seq_140.err.62072
mlp_batchsize_32.out.62055  mlp_sigmoid.out.62061  mlp_width_32.out.61997  rnn_w_att_hist_30.out.62085  rnn_wo_att_seq_140.out.62072
mlp_batchsize_4.err.62053  mlp_tanh.err.62047  mlp_width_32.out.62057  rnn_w_att_hist_40.err.62082  rnn_wo_att_seq_20.err.62084
mlp_batchsize_4.out.62053  mlp_tanh.out.62047  mlp_width_32.out.83064  rnn_w_att_hist_40.out.62082  rnn_wo_att_seq_20.out.62084
mlp_batchsize_64.err.62059  mlp_width_128.err.61996  mlp_width_512.err.62000  rnn_w_att_hist_50.err.62079  rnn_wo_att_seq_40.err.62073
mlp_batchsize_64.out.62059  mlp_width_128.err.62058  mlp_width_512.err.62049  rnn_w_att_hist_50.out.62079  rnn_wo_att_seq_40.out.62073
mlp_depth_1.err.62048  mlp_width_128.err.83062  mlp_width_512.err.83065  rnn_w_att_seq_100.err.62068  rnn_wo_att_seq_60.err.62078
mlp_depth_1.out.62048  mlp_width_128.eval.err.65956  mlp_width_512.eval.err.65955  rnn_w_att_seq_100.out.62068  rnn_wo_att_seq_60.out.62078
mlp_depth_2.err.62050  mlp_width_128.eval.out.65956  mlp_width_512.eval.out.65955  rnn_w_att_seq_120.err.62066  rnn_wo_att_seq_80.err.62080
mlp_depth_2.out.62050  mlp_width_128.out.61996  mlp_width_512.out.62000  rnn_w_att_seq_120.out.62066  rnn_wo_att_seq_80.out.62080
mlp_depth_3.err.62051  mlp_width_128.out.62058  mlp_width_512.out.62049  rnn_w_att_seq_140.err.62067  rnn_wo_att_seq_140.out.62067
mlp_depth_3.out.62051  mlp_width_128.out.83062  mlp_width_512.out.83065  rnn_w_att_seq_140.out.62067
(base) [kpatel48@login02 logs]$
  
```

Figure 58: Screenshot - Log Files

```

(base) [kpatel48@login02 tmp]$ ls
belady      mlp_batchsize_32  mlp_lr_0.00001  mlp_width_32    res.log.gpu06    res.log.gpu26    rnn_w_att_seq_140  rnn_wo_att_seq_20
lru          mlp_batchsize_4   mlp_lr_0.001    mlp_width_512   res.log.gpu08    rnn_w_att_hist_10  rnn_w_att_seq_20   rnn_wo_att_seq_40
LSTM_abi_rank mlp_batchsize_64  mlp_lr_0.1      mlp_width_64    res.log.gpu09    rnn_w_att_hist_20  rnn_w_att_seq_40   rnn_wo_att_seq_60
LSTM_abi_reuse mlp_depth_1       mlp_ReLU        res.log.gpu01    res.log.gpu10    rnn_w_att_hist_30  rnn_w_att_seq_60   rnn_wo_att_seq_80
LSTM_baseline mlp_depth_2       mlp_sigmoid     res.log.gpu02    res.log.gpu11    rnn_w_att_hist_40  rnn_w_att_seq_80   tmp05_10.u8
LSTM_byte_embedder mlp_depth_3      mlp_tanh        res.log.gpu03    res.log.gpu12    rnn_w_att_hist_50  rnn_wo_att_seq_100 tmpfna00g46
mlp_batchsize_1 mlp_depth_4       mlp_width_128   res.log.gpu04    res.log.gpu13    rnn_w_att_seq_100  rnn_wo_att_seq_120 tmpkx8xsrxh
mlp_batchsize_16 mlp_leakyReLU     mlp_width_256   res.log.gpu05    res.log.gpu15    rnn_w_att_seq_120  rnn_wo_att_seq_140 tmp_pbffj_vm
(base) [kpatel48@login02 tmp]$

```

Figure 59: Screenshot - Tmp Files

```

LSTM_baseline.err.62083  mlp_lr_0.00001.err.62063  mlp_width_256.eval.out.65952  mlp_width_64.eval.out.65953  rnn_w_att_seq_60.err.62081
LSTM_baseline.out.62083  mlp_lr_0.00001.out.62063  mlp_width_256.out.61998      mlp_width_64.out.61999      rnn_w_att_seq_60.out.62081
LSTM_byte_embedder.err.62086  mlp_lr_0.001.err.62069  mlp_width_256.out.62054      mlp_width_64.out.62045      rnn_w_att_seq_80.err.62070
LSTM_byte_embedder.out.62086  mlp_lr_0.001.out.62069  mlp_width_256.out.83063      mlp_width_64.out.83066      rnn_w_att_seq_80.out.62070
mlp_batchsize_16.err.62080  mlp_lr_0.1.err.62065     mlp_width_32.err.61997      rnn_w_att_hist_10.err.62077  rnn_wo_att_seq_100.err.62076
mlp_batchsize_16.out.62080  mlp_lr_0.1.out.62065     mlp_width_32.err.62057      rnn_w_att_hist_10.out.62077  rnn_wo_att_seq_100.out.62076
mlp_batchsize_1.err.62056  mlp_ReLU.err.62052       mlp_width_32.err.83064      rnn_w_att_hist_20.err.62075  rnn_wo_att_seq_120.err.62074
mlp_batchsize_1.out.62056  mlp_ReLU.out.62052       mlp_width_32.eval.err.65954  rnn_w_att_hist_20.out.62075  rnn_wo_att_seq_120.out.62074
mlp_batchsize_32.err.62055  mlp_sigmoid.err.62061    mlp_width_32.eval.out.65954  rnn_w_att_hist_30.err.62085  rnn_wo_att_seq_140.err.62072
mlp_batchsize_32.out.62055  mlp_sigmoid.out.62061    mlp_width_32.out.61997      rnn_w_att_hist_30.out.62085  rnn_wo_att_seq_140.out.62072
mlp_batchsize_4.err.62053  mlp_tanh.err.62047       mlp_width_32.out.62057      rnn_w_att_hist_40.err.62082  rnn_wo_att_seq_20.err.62084
mlp_batchsize_4.out.62053  mlp_tanh.out.62047       mlp_width_32.out.83064      rnn_w_att_hist_40.out.62082  rnn_wo_att_seq_20.out.62084
mlp_batchsize_64.err.62059  mlp_width_128.err.61996  mlp_width_512.err.62000     rnn_w_att_hist_50.err.62079  rnn_wo_att_seq_40.err.62073
mlp_batchsize_64.out.62059  mlp_width_128.err.62058  mlp_width_512.err.62049     rnn_w_att_hist_50.out.62079  rnn_wo_att_seq_40.out.62073
mlp_depth_1.err.62048      mlp_width_128.err.83062  mlp_width_512.err.83065     rnn_w_att_seq_100.err.62068  rnn_wo_att_seq_60.err.62078
mlp_depth_1.out.62048      mlp_width_128.eval.err.65956  mlp_width_512.eval.err.65955  rnn_w_att_seq_100.out.62068  rnn_wo_att_seq_60.out.62078
mlp_depth_2.err.62050      mlp_width_128.eval.out.65956  mlp_width_512.eval.out.65955  rnn_w_att_seq_120.err.62066  rnn_wo_att_seq_80.err.62080
mlp_depth_2.out.62050      mlp_width_128.out.61996  mlp_width_512.out.62000     rnn_w_att_seq_120.out.62066  rnn_wo_att_seq_80.out.62080
mlp_depth_3.err.62051      mlp_width_128.out.62058  mlp_width_512.out.62049     rnn_w_att_seq_140.err.62067  rnn_wo_att_seq_140.out.62067
mlp_depth_3.out.62051      mlp_width_128.out.83062  mlp_width_512.out.83065     rnn_w_att_seq_140.out.62067  rnn_wo_att_seq_140.out.62067
(base) [kpatel48@login02 logs]$ cd ..
(base) [kpatel48@login02 kpatel48]$ cd eval/
(base) [kpatel48@login02 eval]$ ls
LSTM_abi_rank      mlp_0.1_lr      mlp_depth_1     mlp_sigmoid     mlp_width_64    rnn_w_attn_60    rnn_w_attn_hist_50  rnn_wo_attn_60
LSTM_abi_reuse     mlp_16_batchsize mlp_depth_2     mlp_tanh        rnn_w_attn_100  rnn_w_attn_80   rnn_wo_attn_100    rnn_wo_attn_80
LSTM_baseline      mlp_1_batchsize mlp_depth_3     rnn_w_attn_120  rnn_w_attn_hist_10  rnn_wo_attn_120  rnn_wo_attn_120
LSTM_byte_embedder mlp_32_batchsize mlp_depth_4     mlp_width_256   rnn_w_attn_140  rnn_w_attn_hist_20  rnn_wo_attn_140
mlp_0.00001_lr     mlp_4_batchsize mlp_leakyReLU   mlp_width_32    rnn_w_attn_20   rnn_w_attn_hist_30  rnn_wo_attn_20
mlp_0.001_lr       mlp_64_batchsize mlp_ReLU        mlp_width_512   rnn_w_attn_40   rnn_w_attn_hist_40  rnn_wo_attn_40
(base) [kpatel48@login02 eval]$

```

Figure 60: Screenshot - Eval Files